LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-CONF-655932

Adaptive Parallelism: Integrated Performance, Power, and Resilience Modeling

D. Li, E. A. Leon, B. R. de Supinski

June 16, 2014

Workshop on Modeling & Simulation of Systems and Applications
Seattle, WA, United States
August 13, 2014 through August 14, 2014

Disclaimer

Adaptive Parallelism: Integrated Performance, Power, and Resilience Modeling

Dong Li
Oak Ridge National Laboratory
lid1@ornl.gov

Edgar A. León Bronis R. de Supinski
Lawrence Livermore National Laboratory*
{leon,bronis}@llnl.gov

The Need for Integrated Modeling

From embedded devices to future exascale computers, increased parallelism in both the number of processing units and nodes will create unprecedented challenges to achieve the expected levels of application performance. System power consumption will be a major consideration. For large-scale systems, failures proportional to the size of the system will impair system usability. To address these challenges, modeling methods must evolve and consider the combined effects of performance, power, and resilience (PPR). Further, modeling methods must be rapid and accurate to evaluate the dynamic tradeoffs posed by PPR in large-scale systems.

Many current modeling methods employ analytical models or architectural, highly-accurate simulators. Analytical models tend to focus on a specific dimension of performance, power, and resilience but often miss the combined interdependent effects. Highly accurate simulators, on the other hand, are not scalable. Further, the overhead imposed by many modeling tools is too high to be used at runtime. Thus, we need more efficient and unified modeling frameworks, which will enable runtime systems to find, in real-time, an efficient operating point in terms of PPR for an application.

In this paper, we propose an infrastructure for modeling parallelism and its combined effects on performance, power, and resilience. Managing and optimizing parallelism dynamically is at the core of meeting the challenging requirements of PPR imposed by future systems. The inherent parallelism of scientific applications varies across execution phases [13]. Matching the degree of parallelism (*parallel configuration*) for an application has complex PPR implications [3, 4, 10, 11]. Our goal is to develop a model-driven approach based on hardware resource utilization that will guide the selection and adaptation of parallel configurations.

Adaptive Parallelism Framework

We base our proposed modeling methodology on two observations. First, performance, power, and resilience have first-order or second-order correlations with hardware component utilization. From a performance perspective, our work reveals, for example, that the number of accesses to the memory hierarchy and the number of executed instructions serve as strong indicators of performance with various levels of parallelism [10, 11, 22, 23]. Power consumption is related to hardware usage intensity [7, 8, 11, 15]. Resilience is related to both application execution time and number of hardware accesses. Given hardware failure rates for specific hardware components, longer application execution times and more hardware accesses expose the application to more random occurrences of hardware failures (including both hard and soft errors). We introduce a new metric, the *vulnerability factor* (VF), which is defined as a function of execution time, number of hardware accesses driven by application characteristics, and component failure rate, to quantify application vulnerability. Thus, resilience, like performance and power, is related to hardware component utilization.

Second, given a parallel region, PPR and thread-level parallelism are strongly correlated statistically. Thus, based on hardware components utilization collected from a few samples of parallel configurations, we can predict PPR for other parallel configurations. We call these representative samples *seminal configurations*.

Based on these observations, we can construct an integrated, PPR model in two phases: offline model training and online model selection. Model training uses machine learning to determine the hardware component utilization information that is most correlated with PPR. The information should be measurable with lightweight hardware counters. Also, seminal configurations are chosen empirically. Empirical observations reveal that PPR data with different levels of parallelism can be clustered into different groups. The parallel configuration that is the *closest* to the center of each group is chosen as a seminal configuration. During offline training, we build a series of PPR models to capture diverse hardware features and application characteristics. Using a diverse set of applications and benchmarks during training is key to producing accurate models. During online model selection, we use a few sample iterations of parallel regions to execute with seminal configurations and collect hardware component utilization information in order to identify the model to use.

*Prepared by LLNL under Contract DE-AC52-07NA27344. LLNL-CONF-655932.

Based on the above modeling methodology, we can accurately predict PPR at runtime for any untested parallelism configuration with low overhead. We enable accurate on-line modeling by building the model offline using a diverse set of application characteristics. Using the resulting models, a runtime system can use high-level policies that indicate the desired levels of performance, power, and resilience. For example, minimizing the vulnerability factor at a marginal performance and power cost; and achieving the best performance and resilience within a power cap. Figure 1 provides an overview of our framework's model construction and deployment.

Our previous investigations show that this methodology can predict performance and power with high accuracy on many-core platforms. These results are encouraging and we plan to integrate our proposed resiliency model into this framework. Looking forward, future milestones include the creation of models for emerging heterogenous memory architectures (multiple levels of memory to provide bandwidth and capacity requirements of future systems), analyzing the effects of data layout and memory parallelism on PPR, and developing new models for heterogenous computing platforms.

Related Work

Performance, power, and resilience modeling and simulation have been studied before, but mostly in isolation. Some related work employs analytical or empirical models to achieve joint optimization of power and performance. For example, Green Queue [17, 24, 25], Adagio [18, 19], and Workload Consolidation [9, 12]. Other work uses detailed hardware analysis for hardware-oriented resilience modeling. For example, Mukherjee et al. [16] define architectural vulnerability factor (AVF) as the probability that a fault in a particular structure will result an error. Biswas et al. [1] show how to compute the AVF of address-based processor structures based on a detailed analysis of architecturally correct execution. Sridharan and Kaeli [20, 21] introduce a new metric to capture the architecture-level fault masking inherent in a program. In addition, fault injection has been widely used to understand application vulnerability [2, 5, 6, 14, 26].

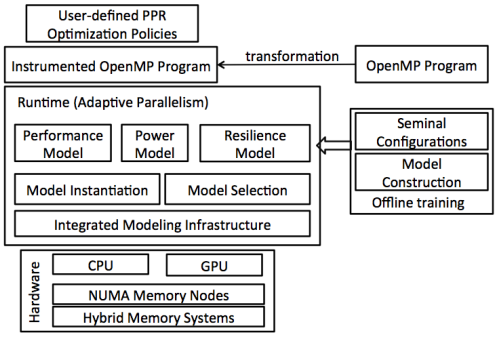

Figure 1: The general framework for adaptive parallelism with integrated PPR modeling

Evaluation of Proposed Methodology

Challenges. Future systems demand modeling and simulation capabilities to help us understand the complex and combined interactions between performance, power, and resilience. Further, modeling and simulation techniques should provide rapid and dynamic evaluation of tradeoffs between them. Our proposed modeling infrastructure is designed to provide lightweight and accurate PPR predictions based on adaptive parallelism. It can be used by runtime systems to manage system resources for a specific set of objectives based on thread-level and memory-level parallelism.

Maturity. Our previous investigations show that our proposed methodology can provide accurate and lightweight modeling of performance and power for OpenMP parallel regions on several multicore architectures [10–12, 22, 23]. We have successfully applied machine-learning techniques to this area to address the challenges associated with an extremely large space of optimizations. This infrastructure provides a strong basis for integrating modeling of different objective functions. Adding modeling capabilities for resilience and reliability along with processor and memory heterogeneity will undoubtedly present significant challenges.

Uniqueness. A key feature of our modeling methodology is our focus on parallelism, which is the central consideration in managing the tradeoffs between power, performance, and resilience. In addition, we will use machine-learning techniques to create multi-dimensional PPR models. The goal of our modeling infrastructure is to guide a runtime system to determine the *right* level of concurrency to achieve desired optimization objectives.

Novelty. Our modeling methodology reveals statistical correlation between measurable hardware events, application characteristics, and PPR. The unique set of features and opportunities provided by our models provide fast exploration of PPR to achieve multi-dimensional optimization.

Applicability. The proposed PPR models have been deployed in an OpenMP runtime to optimize performance and energy efficiency by using adaptive parallelism. Our thesis is that this approach can be successfully applied to other areas such as modeling of emerging memory systems.

Effort. Key milestones include developing models of resilience and memory-level parallelism and investigating their interactions with other objectives such as power and performance. In addition, we need to investigate the accuracy and overhead of our modeling methodology on a variety of hardware resources including homogenous and heterogeneous processor architectures, and emerging heterogeneous memory architectures (multi-level memories).

References

[1] Arijit Biswas, Paul Racunas, Joel Emer, Shubhendu S. Mukherjee, and Ram Rangan. Computing Architectural Vulnerability Factors for Address-Based Structures. In *Intertional Symposium on Computer Architecture*, 2005.

[2] Marc Casas, Bronis R. de Supinski, Greg Bronevetsky, and Martin Schulz. Fault Resilience of the Algebraic Multigrid Solver. In *International Conference on Supercomputing*, 2012.

[3] Matthew Curtis-Maury, James Dzierwa, Christos D. Antonopoulos, and Dimitrios S. Nikolopoulos. Online power-performance adaptation of multithreaded programs using event-based prediction. In *Proceedings of the 20th ACM International Conference on Supercomputing (ICS)*, pages 157–166, Queensland, Australia, June 2006. Acceptance rate: 26%.

[4] Matthew Curtis-Maury, Ankur Shah, Filip Blagojevic, Dimitrios S. Nikolopoulos, Bronis R. de Supinski, and Martin Schulz. Prediction models for multi-dimensional power-performance optimization on many cores. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, pages 250–259, Toronto, Ontario, Canada, October 2008. Acceptance rate: 19%.

[5] Nathan DeBardeleben, Sean Blanchard, Qiang Guan, Ziming Zhang, and Song Fu. Experimental Framework for Injecting Logic Errors in a Virtual Machine to Profile Applications for Soft Error Resilience. In *Euro-par*, 2011.

[6] Siva Kumar Sastry Hari, Sarita V. Adve, Helia Naeimi, and Pradeep Ramachandran. Relyzer: Exploiting Application-Level Fault Equivalence to Analyze Application Resiliency to Transient Faults. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, 2012.

[7] Canturk Isci and Margaret Martonosi. Runtime power monitoring in high-end processors: Methodology and empirical data. In *International Symposium on Microarchitecture*, 2003.

[8] Edgar A. León and Ian Karlin. Characterizing the impact of program optimizations on power and energy for explicit hydrodynamics. In *International Parallel & Distributed Processing Symposium; Workshop on High-Performance, Power-Aware Computing*, HPPAC'14, Phoenix, AZ, May 2014. IEEE.

[9] Dong Li, Surendra Byna, and Scrimat Chakradar. Energy Aware Workload Consolidation on GPU. In *International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems*, 2011.

[10] Dong Li, Bronis R. de Supinski, Martin Schulz, Dimitrios S. Nikolopoulos, and Kirk W. Cameron. Hybrid MPI/OpenMP Power Aware Computing. In *International Parallel and Distributed Processing Symposium (IPDPS)*, 2010.

[11] Dong Li, Bronis R. de Supinski, Martin Schulz, Dimitrios S. Nikolopoulos, and Kirk W. Cameron. Strategies for Energy Efficient Resource Management of Hybrid Programming Models. *Transaction on Parallel and Distributed Systems*, 24(1), 2013.

[12] Dong Li, Dimitrios S. Nikolopoulos, Kirk W. Cameron, Bronis R. de Supinski, and Martin Schulz. Power Aware MPI Task Aggregation Prediction for High End Computing Systems. In *International Parallel and Distributed Processing Symposium (IPDPS)*, 2010.

[13] Dong Li, Jeffrey S. Vetter, Gabriel Marin, Collin McCurdy, Cristi Cira, Zhuo Liu, and Weikuan Yu. Identifying Opportunities for Byte-Addressable Non-Volatile Memory in Extreme-Scale Scientific Applications. In *International Symposium on Parallel and Distributed Processing*, 2012.

[14] Dong Li, Jeffrey S. Vetter, and Weikuan Yu. Classifying soft error vulnerabilities in extreme-scale scientific applications using a binary instrumentation tool. In *SC12: ACM/IEEE International Conference for High Performance Computing, Networking, Storage, and Analysis*, Salt Lake City, 2012.

[15] Tao Li and Lizy Kurian John. Run-time modeling and estimation of operating system power consumption. In *SIGMETRICS*, 2003.

[16] Shubhendu S. Mukherjee, Christopher Weaver, Joel Emer, Steven K. Reinhardt, and Todd Austin. A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor. In *International Symposium on Microarchitecture*, 2003.

[17] Joshua Peraza, Ananta Tiwari, Michael Laurenzano, Laura Carrington, and Allan Snavely. PMaC.s Green Queue: A Framework for Selecting Energy Optimal DVFS Con.gurations in Large Scale MPI Applications. *Concurrency and Computation: Practice and Experience*, 3, 2012.

[18] Barry Rountree, David Lowenthal, Bronis de Supinski, Martin Schulz, Vincent Freeh, and Tyler Bletsch. Adagio: Making DVS Practical for Complex HPC Applications. In *International Conference on Supercomputing (ICS)*, 2009.

[19] Barry Rountree, David Lowenthal, Shelby Funk, Vincent Freeh, Bronis de Supinski, and Martin Schulz. Bounding energy consumption in large-scale MPI programs. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2007.

[20] Vilas Sridharan and David R. Kaeli. Eliminating Microarchitectural Dependency From Architectural Vulnerability. In *International Symposium on High Performance Computer Architecture*, 2009.

[21] Vilas Sridharan and David R. Kaeli. Using PVF Traces to Accelerate AVF Modeling. In *International Workshop on Silicon Errors in Logic - System Effect*, 2010.

[22] ChunYi Su, Dong Li, Dimitrios S. Nikolopoulos, Kirk W. Cameron, Bronis R. de Supinski, and Edgar A. León. Model-Based, Memory-Centric Performance and Power Optimization on NUMA Multiprocessor. In *International Symposium on Workload Characterization*, 2012.

[23] ChunYi Su, Dong Li, Dimitrios S. Nikolopoulos, Mat Grove, Kirk W. Cameron, and Bronis R. de Supinski. Critical Path Based Thread Placement for NUMA Systems. In *International Workshop on Modeling, Benchmarking, and Simulation of High Performance Computing Systems*, 2011.

[24] Ananta Tiwari, Michael Laurenzano, Laura Carrington, and Allan Snavely. Modeling Power and Energy Usage of HPC Kernels. In *International Workshop on High Performance Power-Aware Computing*, 2012.

[25] Ananta Tiwari, Michael Laurenzano, Joshua Peraza, Laura Carrington, and Allan Snavely. Green Queue: Customized Large-scale Clock Frequency Scaling. In *International Conference on Cloud and Green Computing*, 2012.

[26] Xin Xu and Man-Lap Li. Understanding Soft Error Propagation Using Efficient Vulnerability-Driven Fault Injection. In *International Conference on Dependable Systems and Networks*, 2012.